DIGITAL MASS
workshop di cultura digitale *a cura di* ninux.org

Lejos: un progetto opensource per l'unione tra Lego Mindstorms e Java

Daniela Ruggeri

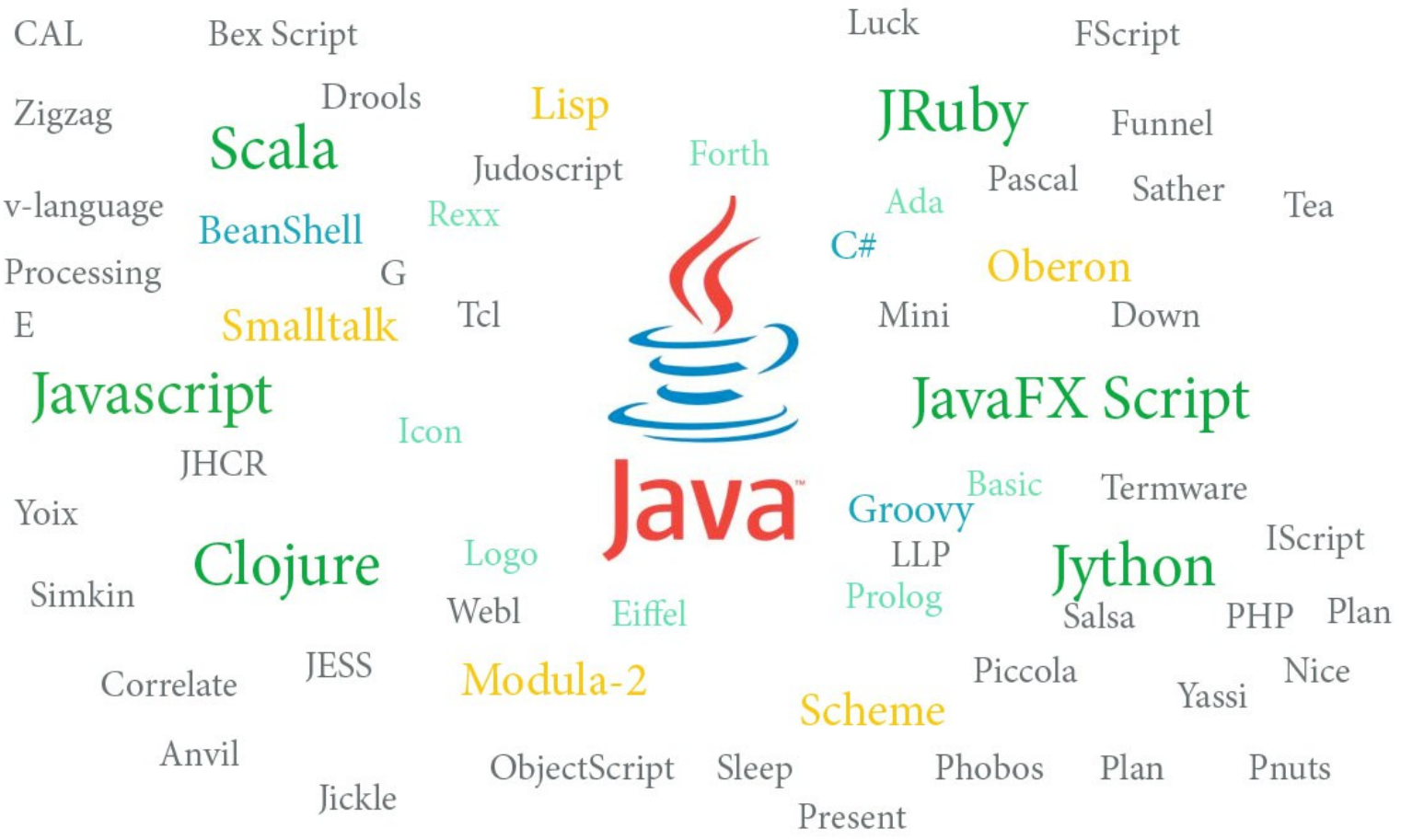
Cos'è Java

- Linguaggio di programmazione ed ambiente di esecuzione progettato da James Gosling, Sun Microsystems
- Le tre anime della piattaforma Java:
  - Java Standard Edition
  - Java Enterprise Edition
  - Java Micro Edition

- Alcune caratteristiche:
  - linguaggio semplice, object-oriented, interpretato, sicuro, multithreaded, portabile, estendibile, orientato agli standard...

Evoluzione di Java

Il linguaggio java oggi è ancora il cardine centrale tra tecnologie che l'hanno precedute, e quelle successive che l'hanno presa come modello



Componibilità

- Avvalersi della componibilità (“comporre”, “mettere insieme”) significa adottare tecnologie costituite da componenti elementari con cui assemblare soluzioni personalizzate
- In questo contesto, componibile è contrapposto a monolitico
- Una delle rivoluzioni del Web 2.0 è la fortissima vocazione alla componibilità di servizi e risorse...

Interoperabilità

- L'interoperabilità è la capacità di una (nuova) tecnologia di integrarsi con soluzioni già esistenti, traendo da esse massimo vantaggio

- Interoperare significa avvalersi di due risorse preziosissime:
  - ciò che è già stato fatto
  - ciò che altri stanno facendo in questo momento

- Ieri era “inutile” reinventare la ruota: oggi non c'è il tempo per farlo!

Buone regole per l'interoperabilità

- L'interoperabilità passa attraverso precise scelte di campo stragegiche:
  - documentare dettagliatamente le interfacce pubbliche delle proprie funzionalità hw/sw
  - utilizzare le risorse ottimali per il campo applicativo oggetto dello sviluppo
  - adottare (e rispettare!) protocolli standard
  - aprirsi alla collaborazione, condividendo conoscenza e risorse
  - coltivare il senso di community, indispensabile per dare/trovare aiuto e percepire le direzioni del settore della propria attività

LEGO & Java

- LEGO e Java sono due esempi brillanti di componibilità e interoperabilità:
  - LEGO non ha bisogno di presentazioni! :-) :
    - C: qualsiasi set di mattoncini può essere utilizzato per arricchire e completare un modello in costruzione
    - I: protocollo di comunicazione e interfacce hardware sono ben specificate e documentate
  - JAVA:
    - C: le tre edizioni condividono quasi totalmente il linguaggio e le librerie possono essere agevolmente portate su architetture e piattaforme diverse
    - I: è disponibile il supporto a tutti i database, framework di comunicazione wired e wireless, web e streaming, datatype, imaging...

LEGO Mindstorms RCX

- Mindstorms Robotic Command eXplorer
  - Sviluppato in collaborazione con il MIT, l'RCX è il primo esempio di “intelligent brick” totalmente programmabile
  - CPU Hitachi 8-bit (16 Mhz)
  - 16 Kb di ROM, 32 Kb di RAM
  - 3 porte di input per i sensori
  - 3 porte di output per i motori
  - un display LCD a 5 caratteri e una porta di comunicazione a infrarossi.

Tool visuale: esempio

1. Il robot deve emettere suoni, alzare il cappello e girare a sinistra quando viene toccata la mano per meno di 3 secondi.

2. Il robot deve emettere suoni diversi alzare il cappello e girare a destra quando viene toccata la mano per più di 3 secondi.

Programma in italiano strutturato

1. → *Se Rilasciato*. Il sensore di contatto del robot parte in posizione premuta, quindi il programma è in attesa che qualcuno tocchi la mano provocando il rilascio del sensore.¶
   1.1. *Condizione verificata*¶
      1.1.1. → *Reimposta timer 1*. Il programma fa partire un timer.¶
      1.1.2. → *Attenti Finché Count 1*. Che significa che il programma attende finché il sensore non viene rilasciato di nuovo.¶
      1.1.3. → *Se timer1 < 3*. Significa se sono passati meno di 3 secondi¶
         1.1.3.1. → *Condizione verificata*.¶
            1.1.3.1.1 → *Bib 3*…Il comando Bip riproduce fino a 6 suoni diversi. In particolare qui è impostato il suono 3.¶
            1.1.3.1.2 → *Ruota sx 1,0 sec.* Il motore collegato alla porta C ruota verso sinistra facendo quindi ruotare il robot a sinistra¶
            1.1.3.1.3 → *Muovi il braccio 2,0 sec.* Il motore collegato alla porta A ruota facendo quindi muovere il braccio 2 secondi sulla testa. Qui non ha importanza la direzione di rotazione¶
            1.1.3.1.4 → *Ruota dx 1,0 sec.* Il motore collegato alla porta C ruota ritorna alla posizione originaria¶
         1.1.3.2. → *Altrimenti*.¶
            1.1.3.2.1 → *Bib 4*…Il comando Bip riproduce fino a 6 suoni diversi. In particolare qui è impostato il suono 4.¶
            1.1.3.2.2 → *Ruota dx 1,0 sec.* Il motore collegato alla porta C ruota verso sinistra facendo quindi ruotare il robot a sinistra¶
            1.1.3.2.3 → *Muovi il braccio 2,0 sec.* Il motore collegato alla porta A ruota facendo quindi muovere il braccio 2 secondi sulla testa. Qui non ha importanza la direzione di rotazione¶
            1.1.3.2.4 → *Ruota sx 1,0 sec.* Il motore collegato alla porta C ruota ritorna alla posizione originaria¶

LEGO Java Operating System

Versioni stabili RCX:

lejos_win32_2_1_0.zip per Windows

lejos_2_1_0.tar.gz per Linux\Mac\Solaris

Versione RCX in test:

lejos.3.0.0-RC2-win32.zip per Windows

lejos.3.0.0-RC2.tar.gz per Linux

scaricabili dal sito http://lejos.sourceforge.net/

DIRECTORY LEJOS

- **bin** contiene gli eseguibili
- **classes** sorgenti java
- **common** contiene i files che sono usati per generare il codice per il linker e la virtual machine.
- **docs** documentazione
- **examples** esempi
- **gameboy_impl** file per la piattaforma Nintendo Game Boy
- **jtools** codice java base per il computer e codice del linker leJOS, usa i file di common/*.db.

DIRECTORY LEJOS

- **rcx_impl** contiene codice C specifico per il firmware RCX.
- **regression** contiene tutta una serie di test di regressione per testare lejOS
- **tools** contiene codice C per creare tools LeJOS
- **unix_impl** Questa directory contiene il codice necessario a creare un tool di emulazione Unix.
- **vmsrc** contiene codice C per la virtual machine.

Librerie LEJOS

- ***rcxrcxcomm.jar*** – gestione sensori, motori e comunicazione da RCX al pc
- ***pcrcxcomm.jar*** – gestione comunicazione da pc a RCX

Packages

Packages	
java.io	Supporto per Input/Output.
java.lang	Classi del linguaggio base Java
java.net	Supporto per Networking/sockets
java.util	Utilità
javax.servlet.http	Classi per gestire un Web Server con lejos
josx.platform.rcx	Classi per gestire sensori RCX, motori RCX ecc.
josx.rcxcomm	Classi per gestire la comunicazione tra RCX e il PC
josx.robotics	Classi e interfacce per gestire azioni e comportamenti dei robot in maniera parametrizzata
josx.util	Altre classi di utilità

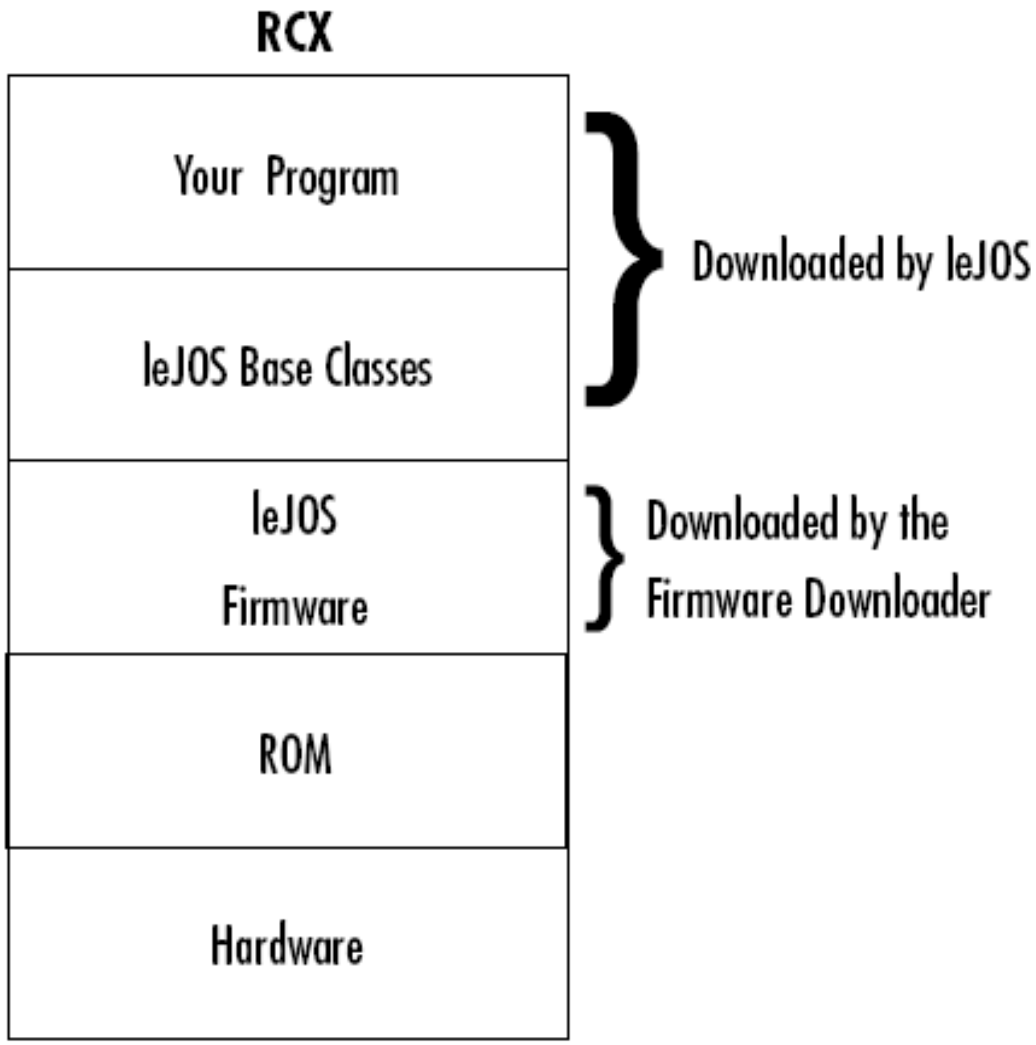
Che cosa ha in meno la 2.1.2

- **Non supporta garbage collection**
- **Non supporta il comando switch**
- **Non supporta tipi primitivi long**
- **La lunghezza massima degli array è 511**.
- ***instanceof*** ritorna *true* per le interfacce. Non permesso sugli array.
- ***java.lang.Class*** non crea istanze (costruttore non applicabile) non si può sincronizzare su metodi statici.
- ***Class.forname*** restituisce un *ClassNotFoundException*, perché non supporta il caricamento dinamico delle classi.
- **Supporta solo fino alla versione 1.4.** Precisamente 1.4.2.x

Che cosa ha in meno la 3.0?

Le stesse funzionalità della versione 2.1.0 però supporta versioni di JDK superiori.

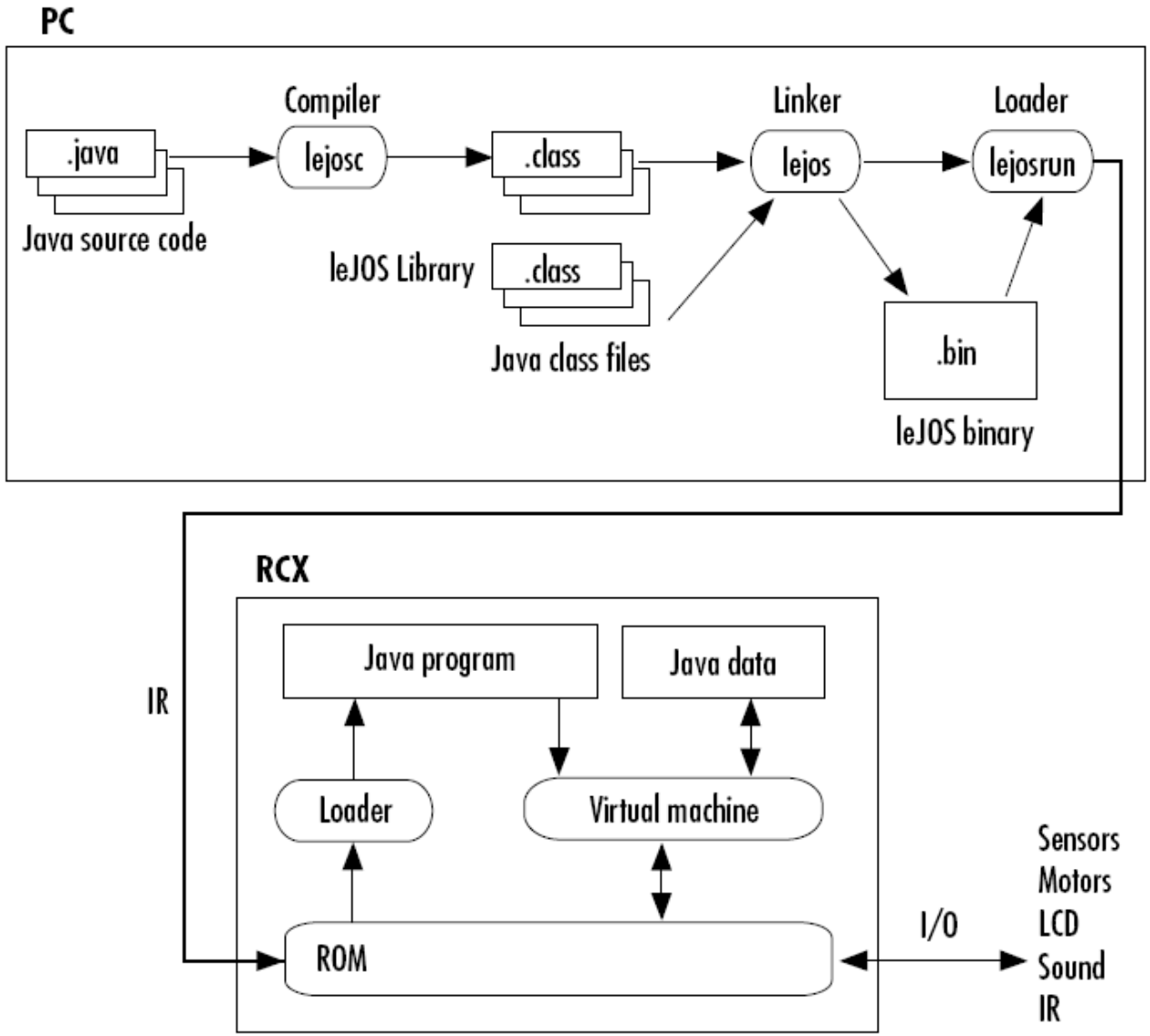
Architettura leJOS



Comandi per RCX

- ***lejosfirmdl*** *(nuovo* ***firmdl****)* fa caricare il firmware in RCX
- ***lejosc*** *(nuovo* ***lejosjc****)* compila i sorgenti java
- ***lejosrun*** trasferisce il programma binario in RCX tramite la torre IR Tower
- ***lejos*** trasforma i programmi in file binari e li trasferisce in RCX tramite la torre IR Tower

Trasferimento codice



Inventorbot (roby)

Specifiche di programma

1. Fare in modo che il robot emetta dei suoni e alzi il cappello quando viene toccata la mano

2. Fare in modo che il robot emetta suoni diversi e giri in sensi diversi a secondo la durata della pressione esercitata sulla mano

3. Fare in modo che il robot saluti se viene colpito da una luce in faccia

Che serve sapere di Java?

- La base del linguaggio
- Gli eventi a delega
- I thread.

Calcolo del tempo

- **josx.util.Timer**

  Oggetto Timer per gestire un intervallo di tempo

- **josx.util.TimerListener**

  Interfaccia ascoltatrice usata con la classe Timer

  In particolare il metodo **timedOut()** dell'interfaccia viene chiamato allo scadere del tempo fissato.

Cambiamento stato sensore

- **osx.platform.rcx.Sensor**

  Astrazione di un sensore. In particolare **Sensor.S1** rappresenta il sensore collegato alla porta 1

- **josx.util.SensorListener**

  Interfaccia ascoltatrice usata con la classe Sensor

  Il metodo stateChanged(Sensor aSource, int aOldValue, int aNewValue) per cambiamento stato del sensore.

- **osx.platform.rcx.SensorCostant**

  Interfaccia usata con la classe Sensor di tutte le costanti riguardanti un sensore

Altre classi usate

- **osx.platform.rcx.Motor**
  Astrazione di un motore. In particolare Motor.A e Motor.C rappresentano i motori collegati alle porte A e C

- **osx.platform.rcx.Button**
  Astrazione di un bottone RCX. In particolare nell'esempio è utilizzato **Button.RUN** ma esistono anche **Button.VIEW** e **Button.PRGM**

- **osx.platform.rcx.Sound**
  Classe usata per emettere suoni

Realizzazione calcolo del tempo

```
// questa variabile vale true se il tempo trascorso è
   minore di 3 secondi
 boolean timeLess3 = true;
/* Creazione istanza della classe Timer associandola
   alla lista ascoltatrice che conterrà
   il tempo che passa */
 Timer timer = new Timer(3000,new TimerListener(){
// trascorsi 3000 millisecondi viene chiamato questo
   metodo
   public void timedOut() {
     timeLess3 = false;
     timer.stop();
   } });
```

Sensori
osx.platform.rcx.SensorCostant

SENSOR_TYPE_LIGHT	Sensore ottico
SENSOR_TYPE_TOUCH	Sensore di contatto
SENSOR_TYPE_TEMP	Sensore di temperatura
SENSOR_TYPE_ROT	Sensore di rotazione
SENSOR_TYPE_RAW	Tutti

SENSOR_MODE_ANGLE	Misura dell'angolo (rotazione)
SENSOR_MODE_BOOL	Booleano true/false
SENSOR_MODE_DEGC	Temperatura in gradi Celsius
SENSOR_MODE_DEGF	Temperature in Fahrenheit
SENSOR_MODE_EDGE	Fa la media delle transazioni e aggunge 1 per ciascun booleano da false a vero e viceversa.
SENSOR_MODE_PCT	Percentuale
SENSOR_MODE_PULSE	Conta le transazioni e aggunge 1 per ciascun booleano da false a vero e viceversa.
SENSOR_TYPE_RAW	Raw valore tra 0 (0V) – 1023 (5V)

Unità di misura Sensori

Volt	Raw	Sensor Ohms	Ottico	Temp. C	Contatto
0.0	0	0	-	-	1
1.1	225	2816	-	70.0	1
1.6	322	4587	100	57.9	1
2.2	450	7840	82	41.9	1
2.8	565	12309	65	27.5	0
3.8	785	32845	34	0.0	0
4.6	945	119620	11	-20.0	0
5.0	1023	Infinito	0	-	0

Stati Sensore di contatto

```
  Sensor.S1.activate(); // attiva il sensore collegato alla porta 1
   Sensor.S1.setPreviousValue (0); // imposta valore iniziale del sensore a 0
(rilasciato)
// imposta tipo di sensore (sensore di contatto) e valori in output (booleani)
   Sensor.S1.setTypeAndMode (SENSOR_TYPE_TOUCH,
SENSOR_MODE_BOOL);
    Sensor.S1.addSensorListener(new SensorListener() {
// questo metodo viene chiamato quando il sensore cambia dallo stato
premuto a quello rilasciato e viceversa.
     public void stateChanged(Sensor aSource,
                    int aOldValue,
                    int aNewValue) {
       if (aOldValue == 1 || aNewValue == 0) {
// sensore in stato di premuto. Start del timer
        timeLess3=true;
        timer.start();
        return;       }
```

Durata meno di 3 secondi

```
Try {
 timer.stop();
   if (timeLess3) {
     Sound.playTone(2100,300);
     Motor.C.forward();
     Thread.sleep(1000);
     Motor.C.stop();
     Sound.beep();
     Motor.A.forward();
     Thread.sleep(2000);
     Motor.A.stop();
     Motor.C.backward();
     Thread.sleep(1000);
     Motor.C.stop();
   }
```

Durata più di 3 secondi

```
else {
        Sound.systemSound(true,3);
         Motor.C.backward();
         Thread.sleep(1000);
         Motor.C.stop();
         Sound.systemSound(true,4);
         Motor.A.backward();
         Thread.sleep(2000);
         Motor.A.stop();
         Motor.C.forward();
         Thread.sleep(1000);
         Motor.C.stop();
         Sound.systemSound(true,5);
         timeLess3=true;
       }
```

Stati Sensore ottico

```
  Sensor.S3.activate(); // attiva il sensore collegato alla porta 3
    Sensor.S3.setPreviousValue (0); // imposta valore iniziale del sensore a 0

// imposta tipo di sensore (sensore ottico) e valori in output (percentuale)
    Sensor.S3.setTypeAndMode (SENSOR_TYPE_LIGHT,
SENSOR_MODE_PCT);

    Sensor.S3.addSensorListener(new SensorListener() {
// questo metodo viene chiamato quando il sensore è colpito dalla luce
      public void stateChanged(Sensor aSource,
                    int aOldValue,
                    int aNewValue) {
                  if (aNewValue > 80) {
                      Motor.A.forward();
                      Thread.sleep(2000);
                       Motor.A.stop();
                   }
```

Comandi DOS per RCX

```
set lejos_home=c:\lejos

set classpath=.;c:\lejos\lib\pcrcxcomm.jar;
   c:\lejos\lib\rcxrcxcomm.jar

set path=c:\jdk1.4.0\bin;c:\lejos\bin;%path%

set RCXTTY=usb

lejosc Inventorbot.java
lejos Inventorbot
```

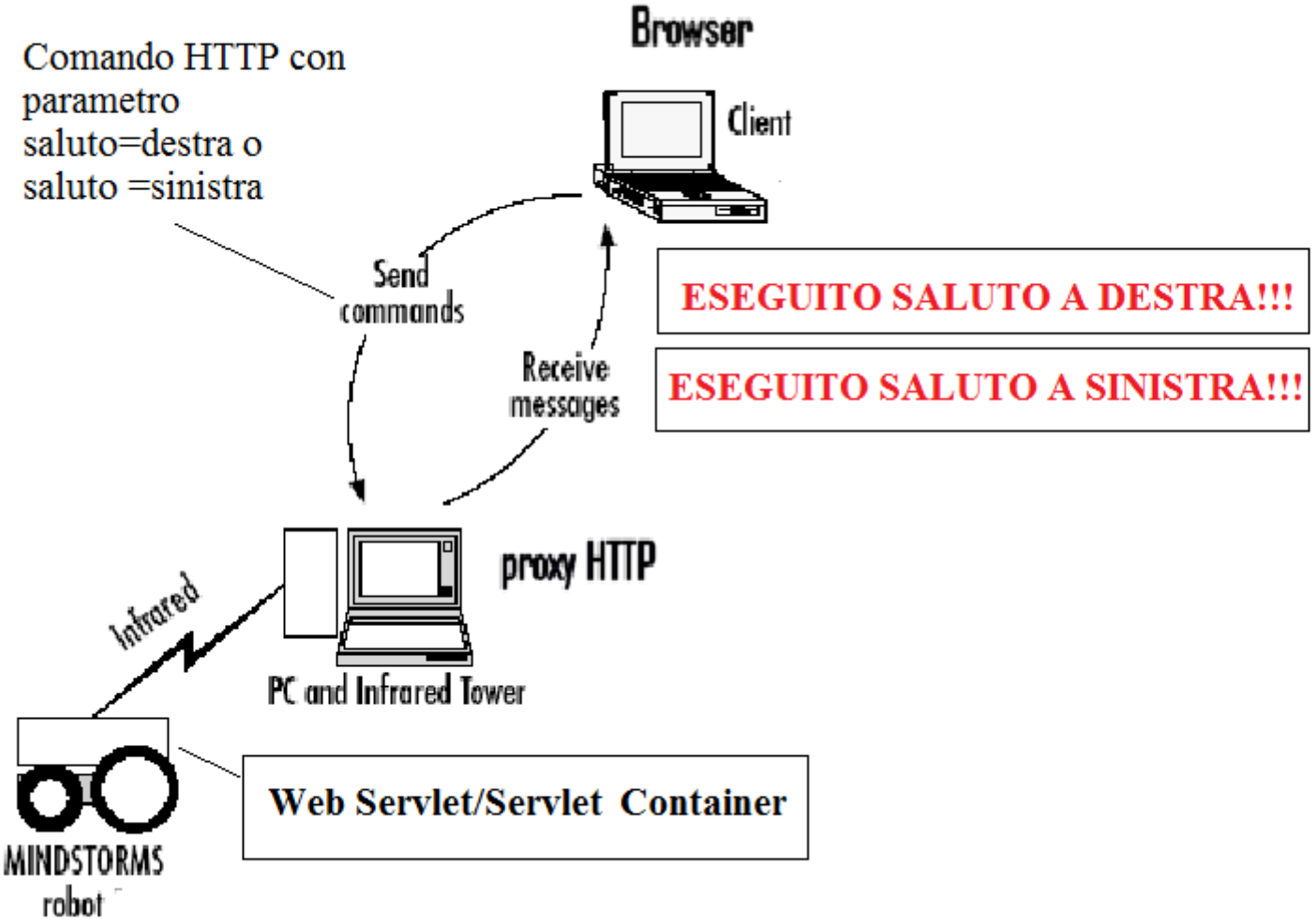
Comandi Linux per RCX

```
#! /bin/bash
set LEJOS_HOME=~/lejos/lejoslejos

CLASSPATH=$LEJOS_HOME/lib/classes.jar:
   $LEJOS_HOME/lib/pcrcxcomm.jar
path=~/lejos_1_0_4alpha/lejos/bin:$path
# driver usb accessibile al link http://legousb.sourceforge.net)
RCXTTY=/dev/usb/legousbtower0
export PATH RCXTTY CLASSPATH LEJOS_HOME

lejosc Inventorbot.java
lejos Inventorbot
```

Comandare l'RCX da web



Specifiche di programma

1. Fare in modo che il robot emetta dei suoni, alzi il cappello e giri a sinistra quando riceve il comando ‘sinistra’
2. Fare in modo che il robot emetta altri suoni, alzi il cappello e giri a destra quando riceve il comando ‘destra’

Alla fine il robot deve rispondere di aver eseguito il comando in linguaggio html.

Che serve sapere di Java?

- linguaggio HTML
- servlet

josx.rcxcomm.HttpProxy

Da shell:

java josx.rcxcomm.HttpProxy -port <portaproxy>

Dal browser:

http://<ipserver>:<portaproxy>/<transazione>?var1=valore1&var2=valore2....

Impostazioni iniziali su RCX

```
public class HTTPInventorbot extends HttpServlet
    implements SensorConstants {
  private OutputStream out;
  private static String html1 = "<HTML><BODY><P>";
  private static String html2 = "<FONT color=#ff0000
    size=7>ESEGUITO SALUTO A DESTRA!!!</FONT";
  private static String html3 = "</P></BODY></HTML>";
  private static String html4 = "<FONT color=#ff0000
    size=7>ESEGUITO SALUTO A SINISTRA!!!</FONT>";
  private static String type = "text/html";
```

public void doGet() (ricezione)

```
out = response.getOutputStream();
String saluto = request.getParameter("saluto");
        if (saluto != null) {
          try {
            if (saluto.charAt(0)=='s') { // parametro saluto='sinistra'
  tournLeft(); // gira a sinistra
              println(html1);
                 println(html4);
                 println(html3);
            } else {
    tournWrite(); // gira a destra
               println(html1);
                  println(html2);
                  println(html3);
              }
          }
```

Spedizione risposta

```
private void println(String s) throws
  IOException {
   for(int i=0;i<s.length();i++)
        out.write((byte) s.charAt(i));
 }
```

ESEMPIO

Da shell:

```
set lejos_home=f:\Progetti1\lejos
set
    classpath=.;f:\Progetti1\lejos\lib\pcrcxcomm.jar;C:\Progetti\l
    ejos\lib\rcxrcxcomm.jar
set path=f:\Progetti1\lejos\bin;%path%
set RCXTTY=usb
java josx.rcxcomm.HttpProxy -port 9090
```

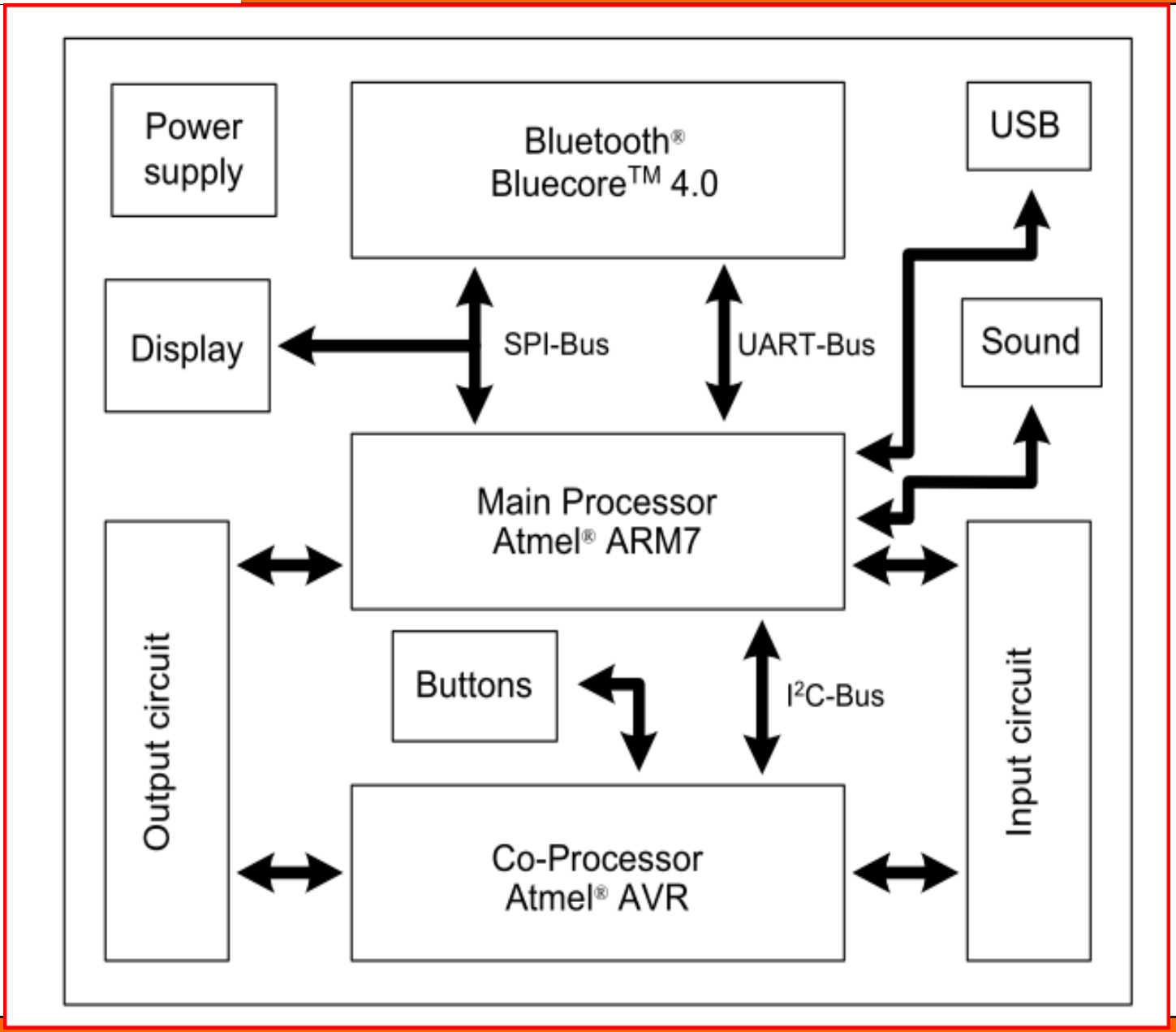
Dal browser:

```
http://localhost:9090/HTTPInventorbot?saluto=sinistra
```

LEGO Mindstorms NXT

- Caratteristiche hardware:
  - Doppio microcontrollore:
    - ARM7 32bit, 256 Kbyte FLASH, 64 Kbyte RAM
    - AVR 8bit, 4 Kbyte FLASH, 512 Byte RAM
  - 4 ingressi, 3 uscite
  - Interfacce USB e Bluetooth
  - Display grafico 100x64px
  - Audio playback 8KHz
  - Possibilità di realizzare network di quattro NXT (master + 3 slave)

Struttura tecnica NXT



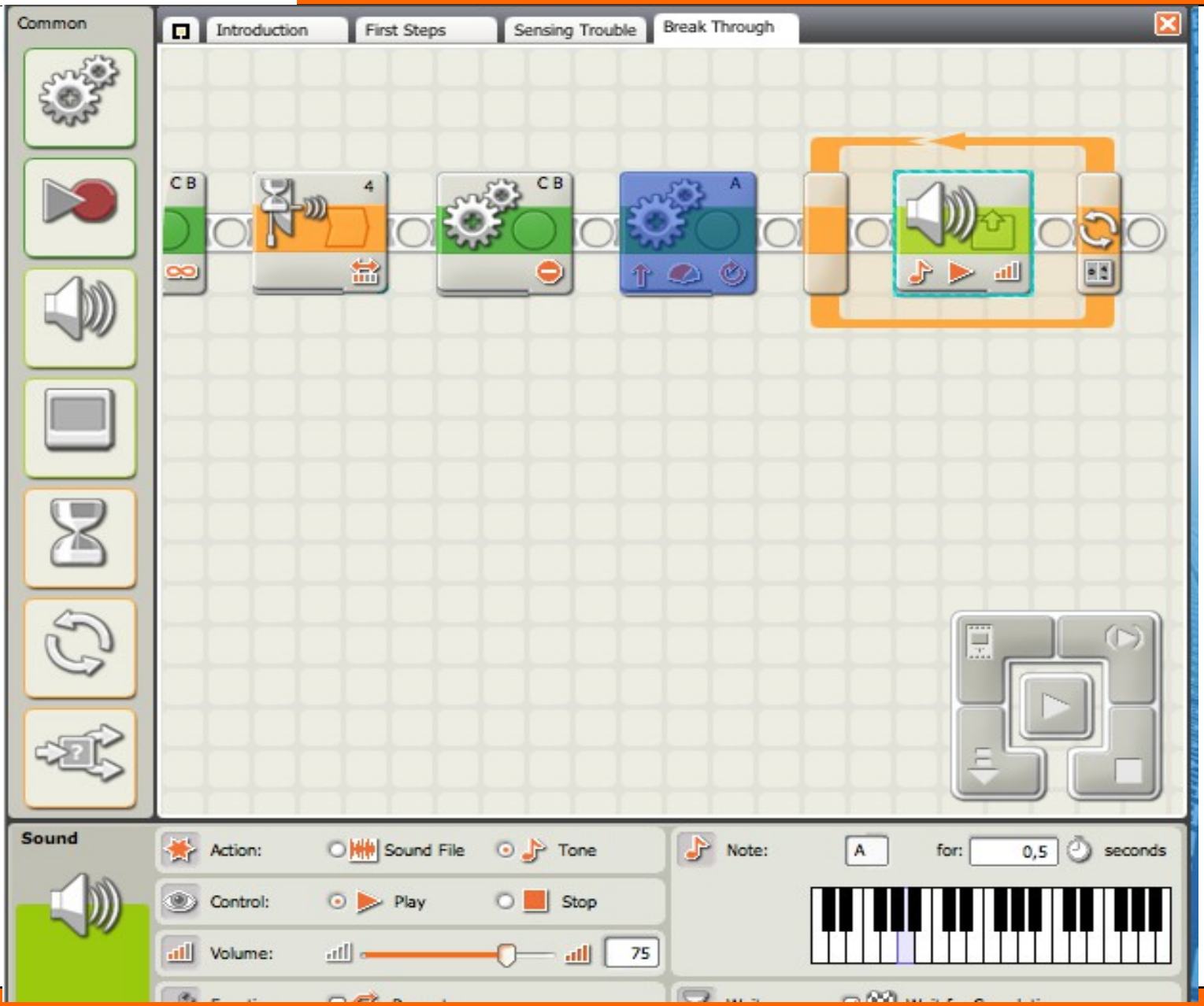
NXT'reme: è tutto open!

- LEGO mette a disposizione della comunità degli sviluppatori risorse per implementare soluzioni originali:
  - Aggiornamenti firmware
  - Documentazione dettagliata dell'interfaccia hardware e delle porte I/O
  - Documentazione dettagliata del protocollo di comunicazione attraverso Bluetooth
  - Strumenti di sviluppo

Sito:

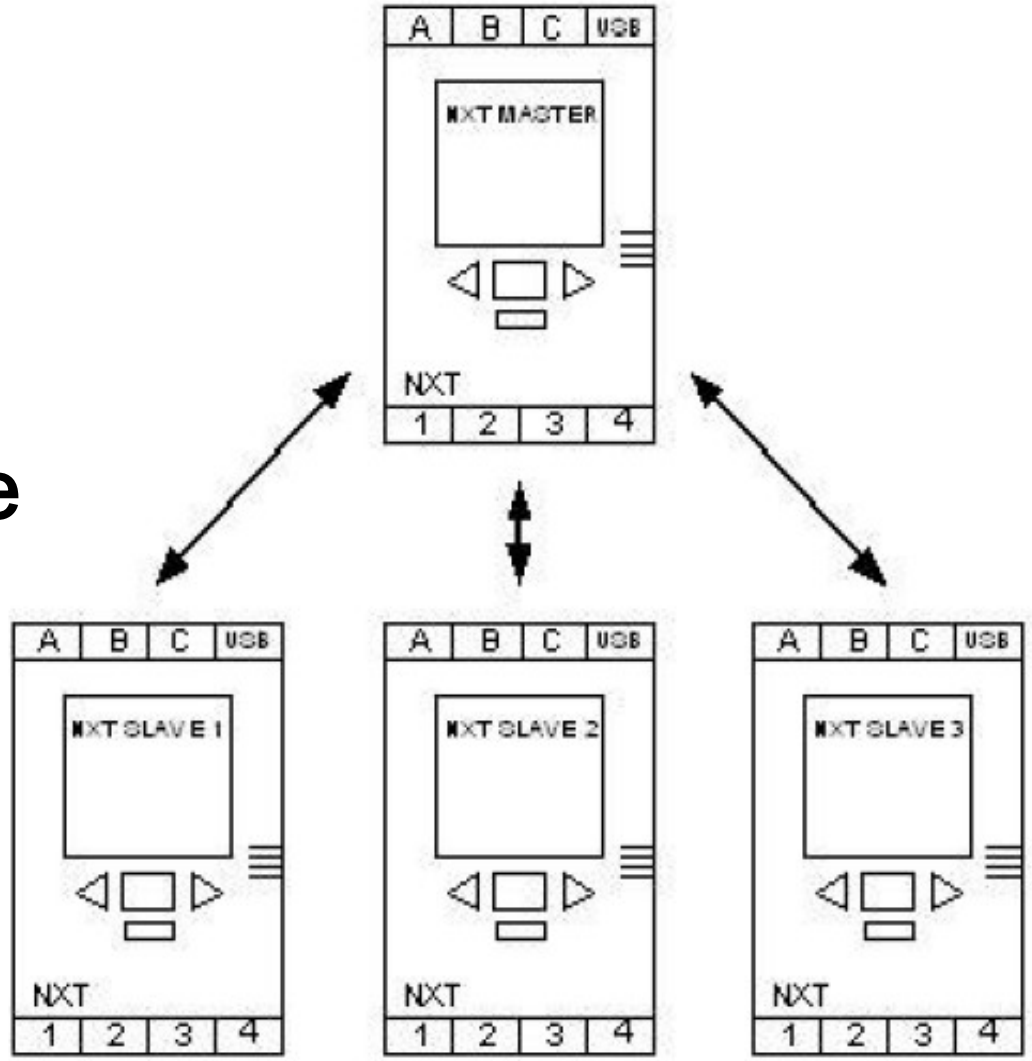
http://mindstorms.lego.com/en-us/whatisnxt/default.aspx

Tool visuale: Labview



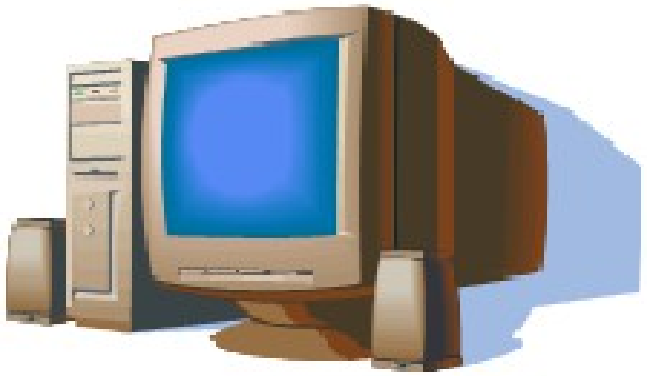
Comunicazione con altri 3 NXT

- Comunicazione Bluetooth
- Un NXT è Master
- Altri NXT sono Slaves (massimo 3)
- 4 canali di comunicazione
- Canale 0 da Slaves a Master
- Canale 1, 2, 3 dal Master verso gli Slaves 1, 2, 3
- Comunicazione 1-1

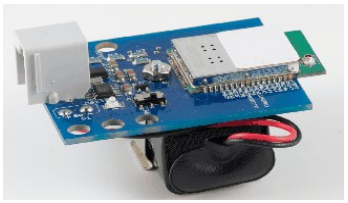


Comunicazione con dispositivi

Bluetooth



Bluetooth

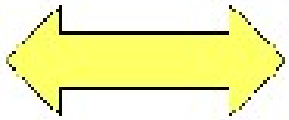


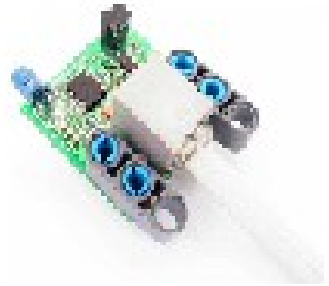
WIFI

Sensore WIFI

Altre forme di comunicazione

Ricevitore Bluetooth
GPS





RCX to NXT
Communication
Adapter (NRLink-Nx)

Tipo di comunicazione da PC

- Verso NXT usando il protocollo LCP (*Lego Communication Protocol*)
  - Con firmware originale NXT nel brick
  - Con firmware Java nel brick
- Verso e da NXT con programmi java
  - Con firmware Java nel brick
  - Programma java di spedizione dati e comandi su PC
  - Programma lejos di ricezione dati e risposta ai comandi nel brick
  - Con protocollo LCP o protocollo creato ad hoc

LEGO Java Operating System

Ultima versione NXJ:

leJOS_NXJ_0.9.1beta-3_win32.zip per **Windows**

leJOS_NXJ_0.9.1beta-3.tar.gz per **Linux\Mac**

scaricabili dal sito http://lejos.sourceforge.net/

Il pacchetto contiene l'installazione Lejos

Contenuto dell'installazione

- Comandi di linea per compilare e per lanciare programmi di supporto
- API Lejos con sorgenti per creare programmi java da installare in NXT
- API Lejos con sorgenti per creare programmi java da installare su PC che comunicano con NXT
- API di altri fornitori per creare programmi java da installare su PC che comunicano con NXT
- Sorgenti di esempi da installare su NXT e su PC
- Driver USB per NXT per piattaforma MAC (libjfantom.jnilib) e per piattaforma Window (jfantom.dll)

Comandi di linea

nxjc	Esegue compilazione programma lejos
nxjlink	Esegue l'assemblazione programma lejos
nxjupload	Esegue l'upload nel NXT
nxj	Esegue l'upload nel NXT e manda in esecuzione il programma
nxjconsole	Permette di connettersi a NXT con bluetooth o USB e di fornire una console visuale ai programmi del brick per la gestione di errori di debug,
nxjflash	Permette di flashare il firmware originale sostituendolo con firmware java
nxjpcc	Esegue compilazione programma su PC
nxjpc	Manda in esecuzione il programma su PC

Programmi di supporto

nxjbrowse	Permette di connettersi a NXT con bluetooth o USB e di eseguire upload, cancellazione e lancio di programmi direttamente nel brick
nxjconsoleviewer	Versione grafica di nxjconsole
nxjflashg	Versione grafica di nxjflash
nxjmonitor	Permette di connettersi a NXT con bluetooth, e visualizzare i messaggi di tracing nel programma NXT attivati dalla classe ***LCPBTResponder***
nxjdataviewer	Permette di connettersi a NXT con bluetooth o USB e visualizzare l'output proveniente dalla classe ***DataLogger*** del programma NXT

Librerie lejos

classes.jar	API Lejos per i programmi da far girare in NXT Disponibili anche i sorgenti
pccomm.jar	API Lejos per i programmi che girano su PC Disponibili anche i sorgenti
pctools.jar	Classi relativi ai programmi di supporto Disponibili anche i sorgenti
charting.jar	Classi per la creazione di grafici per la gestione della Java VM dell'NXT Disponibili anche i sorgenti

Librerie di terze parti

blucove.jar	Implementazione di Java Standard Edition (J2SE) con licenza LGPL JSR-82 che gestisce l'interfaccia Bluetooth su Mac OS X, WIDCOMM, BlueSoleil e Microsoft. Disponibili anche i sorgenti
blucove-gpl.jar	Libreria con licenza GPL che gestisce l'interfaccia Bluetooth su Linux. Disponibili anche i sorgenti
bcel.jar	La Byte Code Engineering Library (Apache Commons BCEL ™) ha lo scopo di offrire agli utenti un modo comodo per analizzare, creare e manipolare file di classe Java binari Disponibili anche i sorgenti
stax-api-1.0.1.jar	Stax è uno standard XML API di elaborazione che consente di trasmettere i dati XML da e per l'applicazione.
jfreechart.jar e jcommon.jar	Insieme di classi java per costruire grafici di livello professionale
common.jar	API del progetto Apache per la gestione e l'analisi delle opzioni e help passati a linea di comando ai programmi java

Tool Java IDE per Lejos

- Plugin per Eclipse <— RACCOMANDATO
- Plugin per Netbeans
- Altri IDE con l'uso di Ant (progetto Apache per la compilazione di script contenuti in file build.xml)

WebNXJ: controllare un robot via web

Specifiche:

- Dalla maschera web far camminare il robot avanti e indietro e fermarlo
- Se il sensore ottico viene colpito da una luce far camminare il robot in avanti
- Se il sensore a ultrasuoni incontra un ostacolo, fermare il robot

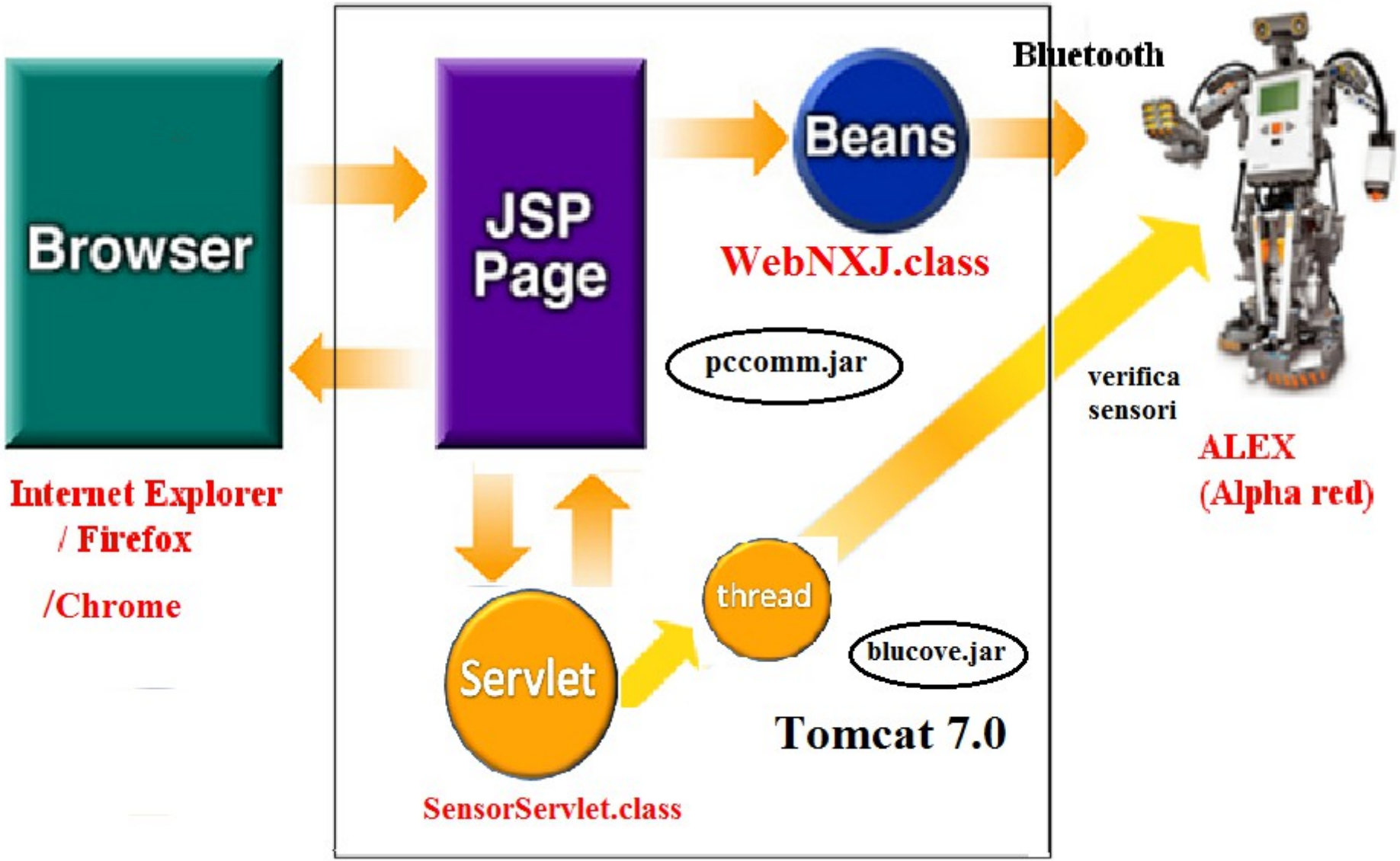
Cosa serve:

- l'ultima versione Java Standard Edition
- un JSP/Servlet container
- una maschera JSP + un Bean Java + una Servlet
- alcune librerie lejos per comunicare con il robot via Bluetooth
- un browser web

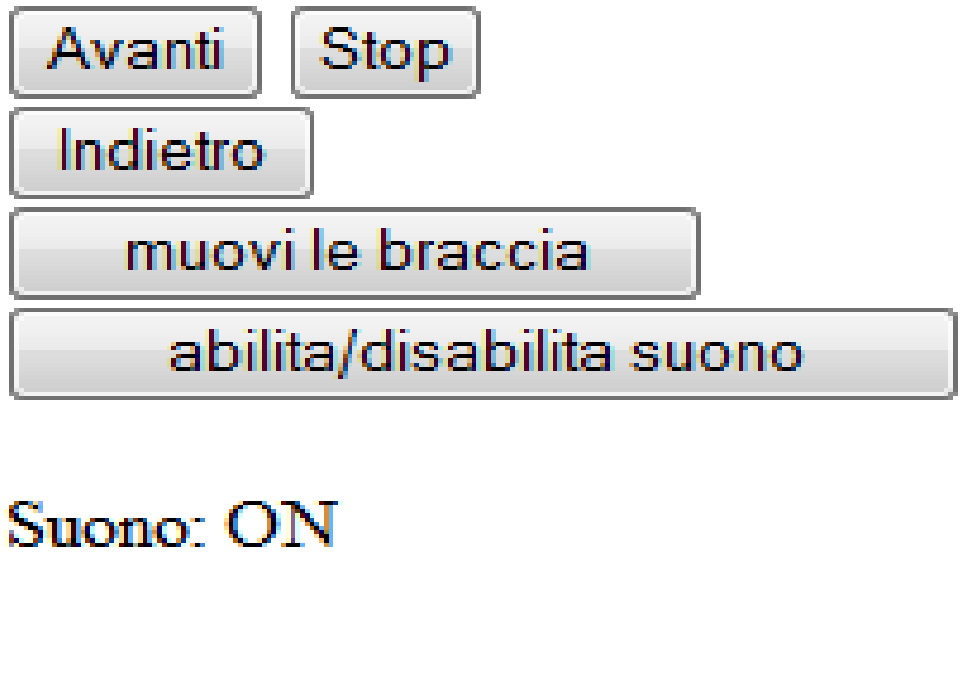
Java Server Pages

- Tecnologia della Sun. Simile a Active Server Pages della Microsoft
- JSP permette di creare pagine HTML e DHTML.
- Utile per costruire pagine che contengono informazioni presenti su altre pagine, prelevate da un database acquisite da input, ecc. (per esempio interagire via bluetooth con un robot)

Architettura



Maschera web



Prodotti usati

- Java Standard Edition *versione 7.*
  http://java.sun.com/javase/downloads/index.jsp
- JSP/Servlet container *Tomcat 7.0*
  http://tomcat.apache.org/
- *Alex.jsp* + il bean *WebNXJ.java*+servlet *SensorServlet.java*
- Librerie *pccomm.jar* e *blucove.jar*
- Il browser *Internet Explorer* o *Firefox* o *Google Chrome*
- Tool IDE *Eclipse* *http://www.eclipse.org*

Cosa è necessario fare

- **Creare collegamento bluetooth con NXT**
- Copiare le librerie pccomm.jar e blucove.jar nella directory \lib di Tomcat
- Eseguire il deploy della webapplication robotic con Alex.jsp, e WebNXJ.class e SensorServlet.class
- Far partire Tomcat
- Caricare la pagina http://localhost:8080/robotic/Alex.jsp

Classi lejos usate

- ***lejos.nxt.remote.NXTCommand*** Classe per la spedizione e ricezione comandi da /per l'NXT
- ***lejos.nxt.Motor*** Astrazione di un motore. In particolare Motor.B e Motor.C rappresentano i motori collegati alle gambe e alle porte B e C e Motor.A il motore collegato alle braccia e alla porta A.
- ***lejos.nxt.SensorPort*** Astrazione della porta di un sensore. Per esempio alla porta **SensorPort.S1** (porta 1) è collegato il sensore a ultrasuoni
- ***lejos.nxt.LightSensor*** Classe per il sensore ottico
- ***lejos.nxt.SoundSensor*** Classe per il sensore suono
- ***lejos.nxt.UltrasonicSensor*** Classe per il sensore a ultrasuoni
- ***lejos.nxt.TouchSensor*** Classe per il sensore di contatto

Classe NXTCommand

Classe che contiene tutti i comandi da/per NXT spediti in questo formato (protocollo LCP)

- Byte 0 = Tipo comando (es. comando *sistema con risposta*)
- Byte 1 = Comando (es. *riproduci un suono contenuto in un file*)
- Byte 4...Byte n (dati informativi e necessari al comando, es. *nome del file contenente il suono da riprodurre*)

Tipi di comandi (NXTProtocol)

Tipo	Byte	Descrizione
DIRECT_COMMAND_REPLY	0x00	Comando diretto con risposta
SYSTEM_COMMAND_REPLY	0x01	Comando di sistema con risposta
REPLY_COMMAND	0x02	Comando di risposta
DIRECT_COMMAND_NOREPLY	0x80	Comando diretto senza risposta
SYSTEM_COMMAND_NOREPLY	0x81	Comando di sistema senza risposta

Comandi di sistema (NXTProtocol)

Tipo	Byte	Descrizione
OPEN_READ	0x80	Apri file in lettura
OPEN_WRITE	0x81	Apri file in scrittura
READ	0x82	Leggi
WRITE	0x83	Scrivi
CLOSE	0x84	Chiude connessione
DELETE	0x85	Cancella file
FIND_FIRST	0x86	Trova il primo file nella directory e con il filtro specificati
FIND_NEXT	0x87	Trova il prossimo file nella directory e con il filtro specificati
GET_FIRMWARE_VERSION	0x88	Ritorna informazioni sul firmware
SET_BRICK_NAME	0x98	Imposta nome NXT
GET_DEVICE_INFO	0x9B	Ottiene informazioni sul dispositivo NXT
DELETE_USER_FLASH	0xA0	Cancella le informazioni nella memory Flash
POLL_LENGTH	0xA1	Ottiene la lunghezza di messaggio di risposta ad un comando
POLL	0xA2	Ottiene il messaggio di risposta ad un comando

Comandi diretti (NXTProtocol)

Tipo	Byte	Descrizione
START_PROGRAM	0x00	Lancia un programma nell'NXT
STOP_PROGRAM	0x01	Ferma un programma nell'NXT
PLAY_SOUND_FILE	0x02	Riproduce un suono di un file
PLAY_TONE	0x03	Riproduce un tono
SET_OUTPUT_STATE	0x04	Imposta parametri per un motore
SET_INPUT_MODE	0x05	Imposta parametri per un sensore
GET_OUTPUT_STATE	0x06	Ottiene parametri di un motore
GET_INPUT_VALUES	0x07	Ottiene parametri di un sensore
RESET_SCALED_INPUT_VALUE	0x08	Azzera impostazioni di un sensore

Comandi diretti (NXTProtocol)

Tipo	Byte	Descrizione
MESSAGE_WRITE	0x09	Spedisce un messaggio ad una casella inbox NXT
RESET_MOTOR_POSITION	0x0A	Azzera contagiri del motore
GET_BATTERY_LEVEL	0x0B	Ritorna il livello della batteria
STOP_SOUND_PLAYBACK	0x0C	Ferma la riproduzione di un suono
KEEP_ALIVE	0x0D	Lascia acceso l'NXT
LS_GET_STATUS	0x0E	Ritorna lo stato del sensore a ultrasuoni
LS_WRITE	0x0F	Imposta informazioni al sensore a ultrasuoni
LS_READ	0x10	Legge informazioni di un sensore a ultrasuoni
GET_CURRENT_PROGRAM_NAME	0x11	Ottiene il nome del programma che sta girando

Es. Ottiene versione firmware

Comando spedito

- Byte 0 = 0x01 tipo comando ***sistema con risposta***
- Byte 1 = 0x88 Comando GET_FIRMWARE_VERSION

Comando ricevuto

- Byte 0 = 0x02 tipo comando ***risposta***
- Byte 1 = 0x88 Comando GET_FIRMWARE_VERSION
- Byte 2 = Codice di ritorno: =**0** – OK ; >**0** KO
- Byte 3 = 0x02 (Versione Minor del protocollo)
- Byte 4 = 0x01 (Versione Major del protocollo)
- Byte 5 = 0x03 (Versione Minor del firmware)
- Byte 6 = 0x01 (Versione Major del firmware)

Versione Protocollo 1.2 e versione firmware 1.3

La pagina Alex.jsp

```
<jsp:useBean id="robotBeanId" scope="session" class="ninux.WebNXJ" />
<h1>Test Alex NXT</h1>
<form method="post" action="Alex.jsp">
<jsp:setProperty name="robotBeanId" property="*" /> <br>
<%String inizio = (String)session.getAttribute("inizio");
        if (inizio == null) {
                inizio="OK";
                session.setAttribute("inizio",inizio);
response.sendRedirect("http://localhost:8080/robotic/SensorServlet");
        }%>
<input type="submit" name="avanti" value="Avanti">
<input type="submit" name="stop" value="Stop"><br>
<input type="submit" name="indietro" value="Indietro"><br>
<input type="submit" name="saluta" value="muovi le braccia"><br>
<input type="submit" name="bottonesuono" value="abilita/disabilita
suono"><br><br>
Suono: <jsp:getProperty name="robotBeanId" property="suono" /> <br>
```

Comando Avanti, Indietro e Stop

```
public void setIndietro(String newValue) throws
Exception {
    System.out.println("indietro: ");
    Motor.B.backward();
    Motor.C.backward();
   }
public void setAvanti(String newValue) throws
Exception {
    System.out.println("avanti: ");
    Motor.B.forward();
    Motor.C.forward();
}
public void setStop(String newValue) throws Exception
{
     System.out.println("stop: ");
     Motor.B.stop();
     Motor.C.stop();
}
```

Comando Muovi le Braccia e abilita/disabita suono

```
public void setSaluta(String newValue) throws Exception
{
      Motor.A.forward();
      Thread.sleep(2000);
      Motor.A.stop();

}
public void setBottonesuono(String newValue) throws
Exception {
      if (suono.equals("ON"))
         suono="OFF";
      else
         suono="ON";
      System.out.println(suono);
}
```

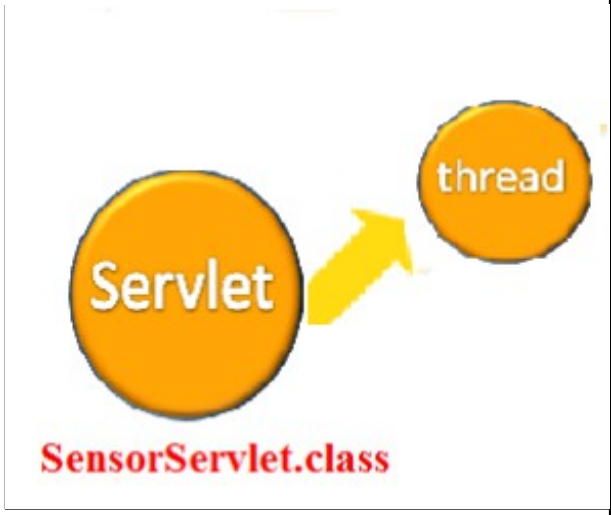
SensorServlet: metodo init()

```
public class SensorServlet extends HttpServlet {
   HttpSession session = null;
// inizializzazione sensori
   LightSensor light;
   SoundSensor sound;
   TouchSensor touch;
   UltrasonicSensor sonic;
public void init() throws ServletException {
super.init();
 sonic = new UltrasonicSensor(SensorPort.S1);
 sound = new SoundSensor(SensorPort.S2);
 touch = new TouchSensor(SensorPort.S3);
 light = new LightSensor(SensorPort.S4);
}
```

SensorServlet: metodo doGet()

```
protected void doGet(HttpServletRequest
request, HttpServletResponse response)
throws ServletException, IOException {

   session = request.getSession();
   (new MioThread(this)).start();
   response.sendRedirect
("http://localhost:8080/robotic/Alex.jsp
");
}
```



SensorServlet: classe interna MioThread

```
public void run() {
        // recupera il bean dalla sessione
   WebNXJ nxj =
(WebNXJ)session.getAttribute("robotBeanId");
   while (true) {
      Try {
// verifica se è stato premuto il sensore di contatto
        if (touch.isPressed()) {
           Sound.beepSequence();
           }
// verifica la presenza di un ostacolo vicino
        if (nxj.suono.equalsIgnoreCase("OFF") &&
sonic.getDistance() <= 30 && Motor.B.isMoving()) {
           nxj.setStop("");
           Sound.buzz();
        }
```

SensorServlet: classe interna MioThread

```
    //verifica l'intensità di una forte luce
          if (light.readValue() > 50)
             nxj.setAvanti("");
    // verifica l'intensità di un forte suono
    if (nxj.suono.equalsIgnoreCase("ON") &&
sound.readValue() > 20 && !Motor.B.isMoving() )
       nxj.setIndietro("");
    }  catch (Exception e1) {
          e1.printStackTrace();
    }
}
```

Evoluzione di Lego Mindstorms

Nella seconda metà del 2013 uscirà

Lego Mindstorms EV3 con le seguenti caratterisiche

- Può essere programmato direttamente nel Brick
- Più memoria e potenza di elaborazione
- Firmware Linux
- Porte USB e slot espansione SD
- Supporto WIFI
- Piena compatibilità con piattafoma IOS e Android
- Manuale di costruzione 17 robot diversi

Link

- http://lejos.sourceforge.net
- http://mindstorms.lego.com/
- http://www.mindsensors.com
- http://www.dexterindustries.com/
- http://www.adrianoparracciani.it/webot

Per informazioni: ruggeridany@yahoo.it

Domande?